

Getting Started Guide

August 28, 2009

Copyright and Permission Notice

Portions copyright (c) 2009 Stanford University and Joy P. Ku
Contributors: Joy P. Ku, Jeanette Schmidt, Peter Eastman

Permission is hereby granted, free of charge, to any person obtaining a copy of this document (the "Document"), to deal in the Document without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Document, and to permit persons to whom the Document is furnished to do so, subject to the following conditions:

This copyright and permission notice shall be included in all copies or substantial portions of the Document.

THE DOCUMENT IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS, CONTRIBUTORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE DOCUMENT OR THE USE OR OTHER DEALINGS IN THE DOCUMENT.

Acknowledgments

OpenMM software and all related activities are funded by the [Simbios](#) National Center for Biomedical Computing through the National Institutes of Health Roadmap for Medical Research, Grant U54 GM072970. Information on the National Centers can be found at <http://nihroadmap.nih.gov/bioinformatics>.

Table of Contents

1	OVERVIEW	6
2	PREREQUISITES.....	7
3	QUICK INSTRUCTIONS.....	8
4	INSTALLING OPENMM.....	9
4.1	Visual Studio 8 version	9
4.2	Visual Studio 9 version	9
4.3	Linux	10
4.4	Mac OSX	11
5	INSTALLING GPU SOFTWARE.....	12
5.1	Installing CUDA for NVIDIA GPUs	12
5.1.1	<i>Windows</i>	<i>12</i>
5.1.2	<i>Mac OS X.....</i>	<i>14</i>
5.2	Installing CAL and Brook for ATI GPUs.....	16
5.2.1	<i>Checking Catalyst Software Suite version</i>	<i>16</i>
5.2.2	<i>Other software required</i>	<i>17</i>
5.2.3	<i>Downloading and installing the Catalyst Software Suite</i>	<i>17</i>
5.2.4	<i>Testing your installation</i>	<i>17</i>
6	VERIFY INSTALLATION BY RUNNING OPENMM EXAMPLE FILES.....	19
6.1	Download Example Files	19
6.2	Run Example Files	19
6.2.1	<i>Visual Studio.....</i>	<i>19</i>
6.2.2	<i>Mac OS X/Linux</i>	<i>21</i>
6.2.3	<i>Troubleshooting</i>	<i>22</i>

1 Overview

OpenMM provides pre-compiled binaries for a number of platforms:

- Windows
 - Visual Studio 8 – supported on NVIDIA and ATI GPUs
 - Visual Studio 9 – supported on NVIDIA GPUs
- Linux (32 and 64 bit) – supported on NVIDIA GPUs
- Mac OS X – supported on NVIDIA GPUs

Source code is also available. This manual provides instructions on using the pre-compiled binaries and running the test examples.

2 Prerequisites

To run OpenMM and the provided test examples, you will need:

- A C++ compiler
 - gcc on Mac/Linux - We have tested the examples on Centos 5.2 with gcc 4.1.2 and on Mac OS X 10.5.6 and 10.5.7 with gcc 4.0.1
 - Visual Studio 8 or 9 on Windows - You can download a free version of Visual C++ 2008 Express Edition (similar to Visual Studio 9) from <http://www.microsoft.com/express/vc/>
- OpenMM pre-compiled binaries for your platform (see Section 4 below)
- OpenMM example files (see Section 6 below)
- To take advantage of the GPU-accelerated molecular dynamics, you must have a supported GPU. You will also need to have the special programming language(s) used for your particular GPU (see Section 5).

3 Quick Instructions

Below is a quick-start guide to getting OpenMM and running the provided test examples. More details follow in the subsequent sections.

1. Download OpenMM binaries from <http://simtk.org/home/openmm>. Extract files and save to C:\ProgramFiles\OpenMM (Windows) or /usr/local/openmm (Mac OS X/Linux).
2. Set path variables for the lib directory within in the openmm or OpenMM folder – See Section 4 for more detailed instructions.
3. Install GPU software, if applicable – See Section 5 for more detailed instructions.
4. Download and unzip OpenMMExample.zip file (available on <http://simtk.org/home/openmm>). These are also included in the *examples* folder when you download the OpenMM source code.
5. Build and run the “HelloArgon” program to test the installation – see Section 6 for more detailed instructions.
 - a. On Linux/Mac OS X, type `make`. Run the “HelloArgon” program.
 - b. On Windows, double click on “HelloArgon.sln” located in the HelloArgonVS8 folder. Make sure the Solution Configuration in Visual Studios is set to "Release"; due to incompatibilities among Visual Studios versions, we do not provide pre-compiled debug binaries. Build the program (Select Debug -> Start Without Debugging).

4 Installing OpenMM

The pre-compiled OpenMM libraries can be obtained from <http://simtk.org/home/openmm>. Click on “Downloads.” Under the list of “Pre-compiled binaries,” select the file that corresponds to your platform. The instructions below focus just on installing from the binaries. However, source code for OpenMM is also available.

4.1 Visual Studio 8 version

Extract all files from the zip file and place them in C:\Program Files \OpenMM. Programs that use OpenMM should include C:\Program Files \OpenMM\lib in the PATH. If you have an ATI GPU, also include C:\Program Files\OpenMM\lib\brook in the PATH. To set the PATH permanently:

1. Click on Start -> Control Panel -> System
2. Click on the “Advanced” tab or the “Advanced system settings” link
3. Click “Environment Variables”
4. Under “System variables,” double-click the line for “Path”
5. Add C:\Program Files \OpenMM\lib (and if appropriate, C:\Program Files\OpenMM\lib\brook) to the “Variable value”
6. If you install OpenMM to a location other than C:\ProgramFiles, you will also need to set the variable OPENMM_PLUGIN_DIR. Under “System variables,” click the “New” button. Set the “Variable name” to OPENMM_PLUGIN_DIR. Set the “Variable value” to the path for the *OpenMM/lib/plugins* directory. Click “OK.”
7. Click “OK”

4.2 Visual Studio 9 version

Extract all files from the zip file and place them in C:\Program Files \OpenMM. Programs that use OpenMM should include C:\Program Files \OpenMM\lib in the PATH. To set the PATH permanently:

1. Click on Start -> Control Panel -> System
2. Click on the “Advanced” tab
3. Click “Environment Variables”
4. Under “System variables,” select the line for “Path”
5. Add C:\Program Files \OpenMM\lib (and if appropriate, C:\Program Files\OpenMM\lib\brook) to the “Variable value”
6. If you install OpenMM to a location other than C:\ProgramFiles, you will also need to set the variable OPENMM_PLUGIN_DIR. Under “System variables,” click the “New” button. Set the “Variable name” to OPENMM_PLUGIN_DIR. Set the “Variable value” to the path for the *OpenMM/lib/plugins* directory. Click “OK.”
7. Click “OK”

4.3 Linux

Extract all files from the zip file and place them in /usr/local/openmm. Programs that use OpenMM should include /usr/local/openmm/lib in the LD_LIBRARY_PATH. To set the LD_LIBRARY_PATH, type :

```
export LD_LIBRARY_PATH=/usr/local/openmm/lib:$LD_LIBRARY_PATH
```

This sets the LD_LIBRARY_PATH only for the terminal you are in. To set it permanently, you will need to add it to, for example, your .bash_profile if you use the BASH shell.

If you choose to install OpenMM some place other than the default location (/usr/local/openmm), you will need to also set the OPENMM_PLUGIN_DIR to the openmm/lib/plugins directory. For example:

```
export OPENMM_PLUGIN_DIR=/usr/jpk/openmm/lib/plugins
```

Again, to set the variable permanently, you will need to add it to, for example, your .bash_profile if you use the BASH shell.

4.4 Mac OSX

Extract all files from the zip file and place them in `/usr/local/openmm`. Programs that use OpenMM should include `/usr/local/openmm/lib` in the `DYLD_LIBRARY_PATH`. To set the `DYLD_LIBRARY_PATH`, type:

```
export DYLD_LIBRARY_PATH=/usr/local/openmm/lib:$DYLD_LIBRARY_PATH
```

This sets the `DYLD_LIBRARY_PATH` only for the terminal you are in. To set it permanently, you will need to add it to your `.bash_profile`.

If you choose to install OpenMM some place other than the default location (`/usr/local/openmm`), you will need to also set the `OPENMM_PLUGIN_DIR` to the `openmm/lib/plugins` directory. For example:

```
export OPENMM_PLUGIN_DIR=/Users/jpk/openmm/lib/plugins
```

Again, to set the variable permanently, you will need to add it to, for example, your `.bash_profile` if you use the BASH shell.

5 Installing GPU software

To take advantage of the GPU acceleration provided via OpenMM, your computer needs to be equipped with one of the supported GPU cards:

Supported NVIDIA GPUs:

http://www.nvidia.com/object/cuda_learn_products.html

Supported ATI GPUs on desktops (Note supported operating systems):

<http://ati.amd.com/technology/streamcomputing/requirements.html>

ATI Mobility Radeon GPU boards on laptops from ATI notebook PC partners

You also need to install CUDA (for NVIDIA GPUs) or CAL and Brook (for ATI GPUs), and test them before running OpenMM and the provided examples.

ATI does not provide CAL and Brook for Macintosh desktops or laptops. However, it may be possible to get the acceleration on a Macintosh desktop with an ATI Radeon GPU by running Windows with Bootcamp (<http://www.apple.com/macosx/features/bootcamp.html>). VMWare, another software for running Windows on a Macintosh, will **not** enable your machine to access the GPU.

5.1 Installing CUDA for NVIDIA GPUs

For NVIDIA GPUs, you need to have CUDA version 2.2 or later installed to get the GPU acceleration. It is recommended that you test your installation before trying to run OpenMM and the provided examples.

5.1.1 Windows

1. Go to http://www.nvidia.com/object/cuda_get.html

2. Download and install the CUDA Driver, the CUDA Toolkit, and the CUDA SDK code samples. We recommend using version 2.2, as some of our machines had driver incompatibility issues with later versions, but later versions might work for your machine. The driver and toolkit are needed to get the GPU acceleration. The code samples are required for testing purposes.
3. To verify that you've installed things correctly, run a sample program available with the SDK code samples.
 - a. Go to Start -> All Programs -> NVIDIA Corporation -> NVIDIA CUDA SDK -> NVIDIA CUDA SDK Browser
 - b. A window appears showing all the different sample programs you can try running (Figure 5.1).
 - c. Locate the program "Device Query" and click on the associated "Run" link on the right-hand side. If things are running correctly, a window will appear stating how many devices are running CUDA (there should be at least 1) and that it/they passed the test.

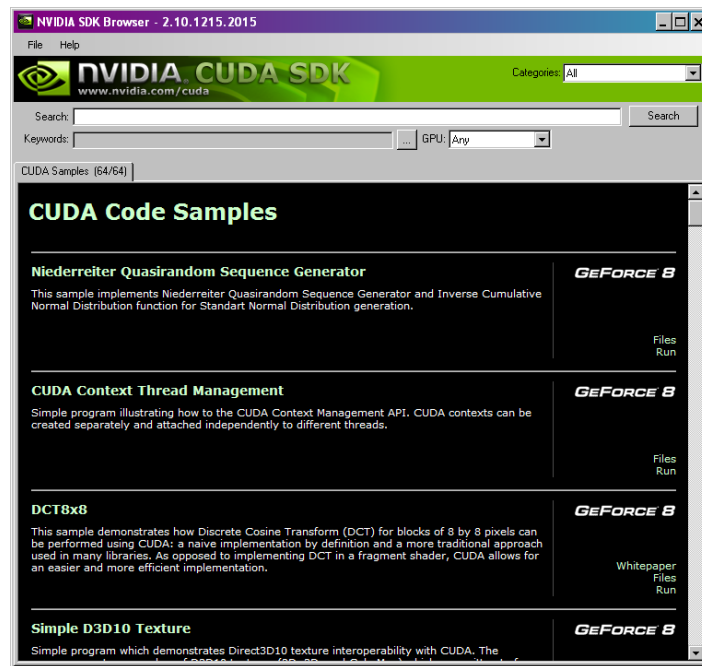


Figure 5.1: Window for browsing the NVIDIA code samples

5.1.2 Mac OS X

1. Go to http://www.nvidia.com/object/cuda_get.html
2. Download and install the CUDA Toolkit and the CUDA SDK code samples, version 2.2. Install the toolkit using a *custom install* and make sure all boxes, including the kernel extension are checked. With version 2.2, the driver is included; with later versions, which may work depending on your setup, the driver is a separate download, and you need to make sure you download and install that as well. The toolkit and driver are needed to get the GPU acceleration. The code samples are required for testing purposes.
3. To verify that you've installed things correctly, run a sample program available with the SDK code samples.
 - a. Open a terminal window. Go to Macintosh HD -> Applications -> Utilities. Click on Terminal.

- b. Set your environment variables so that your computer can locate the CUDA programs by typing:

```
export PATH=/usr/local/cuda/bin:$PATH

export DYLD_LIBRARY_PATH=/usr/local/cuda/lib:
$DYLD_LIBRARY_PATH
```

This sets the environment variables only for the terminal you are in. To set them permanently, you will need to add it to your `bash_profile`.

- c. Within the terminal window, navigate to the location of the code samples. If you installed everything in the default directories, then you would type one of the following (depending on the version of SDK that you downloaded):

```
cd /Developer/CUDA
```

OR

```
cd /Developer/GPU Computing/C
```

- d. Compile the test programs by typing:

```
make
```

- e. Navigate to the location of the compiled programs by typing one of the following (depending on the version of SDK that you downloaded):

```
cd /Developer/CUDA/bin/darwin/release
```

OR

```
cd /Developer/GPU Computing/C/bin/darwin/release
```

- f. Run the `deviceQuery` program:

```
./deviceQuery
```

If things are running correctly, you will see how many devices are running CUDA (there should be at least 1) and a printout saying that it/they passed the test.

Troubleshooting:

If no devices are found, verify that you have a supported GPU card. If you do, re-run the installer and make sure to select a custom installation verifying that all boxes, including the kernel extension, are checked.

If you have multiple GPUs and only one is activated, this may be because of the energy-saving options (this is the case for new MacBook Pros, which ship with a deactivated 9600M GPU). To change the energy-saving options, click System Preferences -> Energy Saver and set the graphics option to “Higher Performance.” You will need to log out and then log back in for the new options to take effect.

Additional instructions and troubleshooting tips are provided in the “Getting Started” manual on the CUDA download site.

5.2 Installing CAL and Brook for ATI GPUs

For ATI GPUs, you must have CAL and Brook installed to get the GPU acceleration. Catalyst Software Suite version 8.12 and later provide these. Again, ATI does not provide this software for Macintosh desktops or laptops.

5.2.1 Checking Catalyst Software Suite version

On Windows, you can check what version of Catalyst Software Suite you have:

- 1) Go to Start -> All Programs -> Catalyst Control Center -> Catalyst Control Center

- 2) In the Catalyst Control Center, click Information Center -> Graphics Software -> Catalyst Version. If you don't see a listing for Catalyst Version, you don't have the driver at all and need to download it. Only versions 8.12 and later have the needed software.

5.2.2 Other software required

Check <http://ati.amd.com/technology/streamcomputing/requirements.html> for the latest information about supported operating systems. In particular, you may need to update your Windows operating system to have the latest service pack (SP).

For Windows, you will also need Microsoft .NET Framework 2.0. To check to see if you already have it:

1. Go to Start -> Control Panel -> Add or Remove Programs
2. Look to see if Microsoft .NET Framework 2.0 is listed in the window that appears. If not, you need to download and install it. The AMD/ATI website provides a link for your specific platform via their on-line installation instructions. You can also go directly to Microsoft to download it:

<http://msdn.microsoft.com/en-us/netframework/aa731542.aspx>

Select the redistributable package that corresponds to your computer: x86 version, x64 (64-bit) version or the IA64 (Intel 64-bit) version.

5.2.3 Downloading and installing the Catalyst Software Suite

To get the Catalyst Software Suite, which contains the versions of CAL and Brook needed, go to: <http://ati.amd.com/support/driver.html>. Make sure you get version 8.12 or later.

5.2.4 Testing your installation

To verify that you've installed things correctly, download and install SDK from <http://ati.amd.com/technology/streamcomputing/sdkdwld.html>.

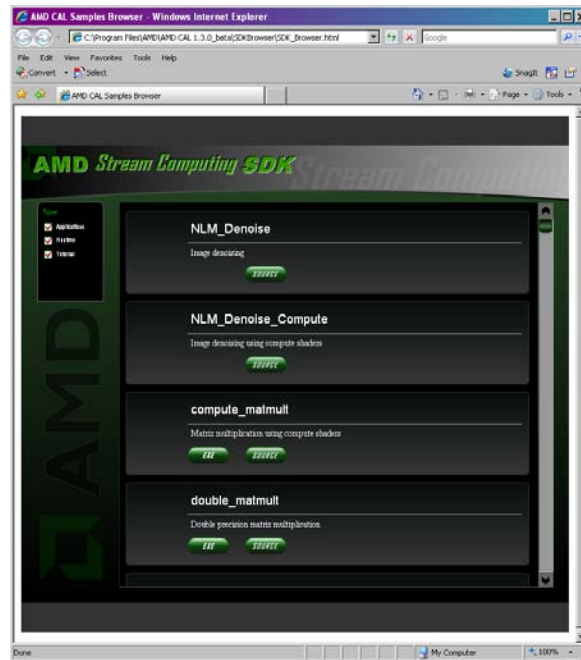


Figure 5.2: Window for browsing the AMD/ATI code samples

1. Go to Start -> All Programs -> AMD -> AMD CAL 1.3.0_beta -> AMD CAL Samples Browser to bring up the window shown in Figure 5.2.
2. Scroll to the bottom of the list and find FindNumDevices. Click on the associated EXE button. When asked if you want to run or save the file, select "Run."
3. If things are running correctly, a window will appear stating how many devices are running CAL (there should be at least 1).

6 Verify Installation By Running OpenMM Example Files

6.1 Download Example Files

Go to <http://simtk.org/home/openmm> and click on “Downloads.” Select the OpenMMExamples.zip file and unzip them to wherever you like. Note: the examples are also included in the *examples* folder when you download the OpenMM source code.

6.2 Run Example Files

Four example files are in the *examples* folder. See the README file in the package for details about the different examples.

The instructions below are for running the HelloArgon program. A similar process would be used to run the other examples.

6.2.1 Visual Studio

Navigate to wherever you saved the example files. Descend into the directory folder HelloArgonVS8. Double-click the file HelloArgon.sln (a Microsoft Visual Studio Solution file). Visual Studio will launch.

Note: these files were created using Visual Studio 8. If you are using Visual Studio 9 (2008 Express Edition), the program will ask if you want to convert the files to the new version. Agree and continue through the conversion process.

In Visual Studio, make sure the "Solution Configuration" is set to "Release" and not "Debug". The "Solution Configuration" can be set using the drop-down menu in the top toolbar, next to the green arrow (see Figure 6.1 below).

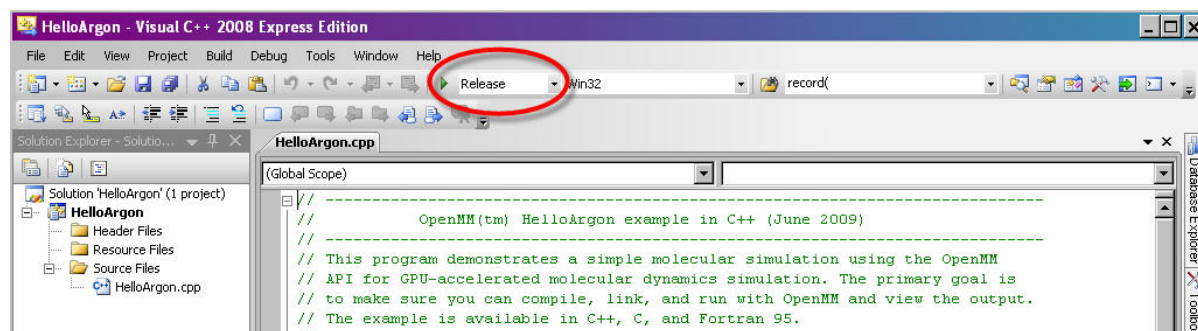


Figure 6.1: Setting "Solution Configuration" to "Release" mode in Visual Studio

From the command options select Debug -> Start Without Debugging.

You should see a series of lines like the following output on your screen:

```
...
MODEL      250
ATOM       1  AR   AR   1      0.233  0.000  0.000  1.00  0.00
ATOM       2  AR   AR   1      5.068  0.000  0.000  1.00  0.00
ATOM       3  AR   AR   1      9.678  0.000  0.000  1.00  0.00
ENDMDL
MODEL      251
ATOM       1  AR   AR   1      0.198  0.000  0.000  1.00  0.00
ATOM       2  AR   AR   1      5.082  0.000  0.000  1.00  0.00
ATOM       3  AR   AR   1      9.698  0.000  0.000  1.00  0.00
ENDMDL
MODEL      252
ATOM       1  AR   AR   1      0.165  0.000  0.000  1.00  0.00
ATOM       2  AR   AR   1      5.097  0.000  0.000  1.00  0.00
ATOM       3  AR   AR   1      9.717  0.000  0.000  1.00  0.00
ENDMDL
```

The very first line of the output will indicate whether you are running on a CPU or a GPU (REMARK Using....). If you have a supported GPU, the program should, by default, run on the GPU.

You can also output the results to a PDB file that could be visualized using programs like VMD (<http://www.ks.uiuc.edu/Research/vmd/>) or PyMol (<http://pymol.sourceforge.net/>).

To do this within Visual Studios:

- 1) Right-click on the project name and select the “Properties” option.
- 2) On the “Property Pages” form, select “Debugging” under the “Configuration Properties” node.
- 3) In the “Command Arguments” field, type:

```
> argon.pdb
```

This will save the output to a file called “argon.pdb” in the current working directory. If you want to save it to another directory, you will need to specify it.

- 4) Select “OK”

Now, when you run the program in Visual Studio, no text will appear. After a long pause, you should see the message “Press any key to continue...,” indicating that the program is complete and that the PDB file has been completely written.

Check the troubleshooting section below if you have problems running the example.

6.2.2 Mac OS X/Linux

Navigate to wherever you saved the example files.

Verify your makefile by consulting the MakefileNotes file in this directory, if necessary.

Type:

```
make
```

Then run the program by typing:

```
./HelloArgon
```

You should see a series of lines like the following output on your screen:

...

```
MODEL      250
ATOM       1  AR   AR       1      0.233  0.000  0.000  1.00  0.00
ATOM       2  AR   AR       1      5.068  0.000  0.000  1.00  0.00
ATOM       3  AR   AR       1      9.678  0.000  0.000  1.00  0.00
ENDMDL
MODEL      251
ATOM       1  AR   AR       1      0.198  0.000  0.000  1.00  0.00
ATOM       2  AR   AR       1      5.082  0.000  0.000  1.00  0.00
ATOM       3  AR   AR       1      9.698  0.000  0.000  1.00  0.00
ENDMDL
MODEL      252
ATOM       1  AR   AR       1      0.165  0.000  0.000  1.00  0.00
ATOM       2  AR   AR       1      5.097  0.000  0.000  1.00  0.00
ATOM       3  AR   AR       1      9.717  0.000  0.000  1.00  0.00
ENDMDL
```

The very first line of the output will indicate whether you are running on the CPU or a GPU (REMARK Using....). If you have a supported GPU, the program should, by default, run on the GPU.

You can also output the results to a PDB file that could be visualized using programs like VMD (<http://www.ks.uiuc.edu/Research/vmd/>) or PyMol (<http://pymol.sourceforge.net/>) by typing:

```
./HelloArgon > argon.pdb
```

6.2.3 Troubleshooting

6.2.3.1 You get the error message: "The application failed to initialize properly (0xc0150002). Click OK to terminate the application"

This may be because OpenMM is trying to run the ATI GPU software (Brook) and is unable to locate it. You do not need to have this software installed to run the examples. Just move all Brook-related files located in the OpenMM/lib/plugins directory (file names will contain the word "Brook" in them) to a higher level directory and run the example again.